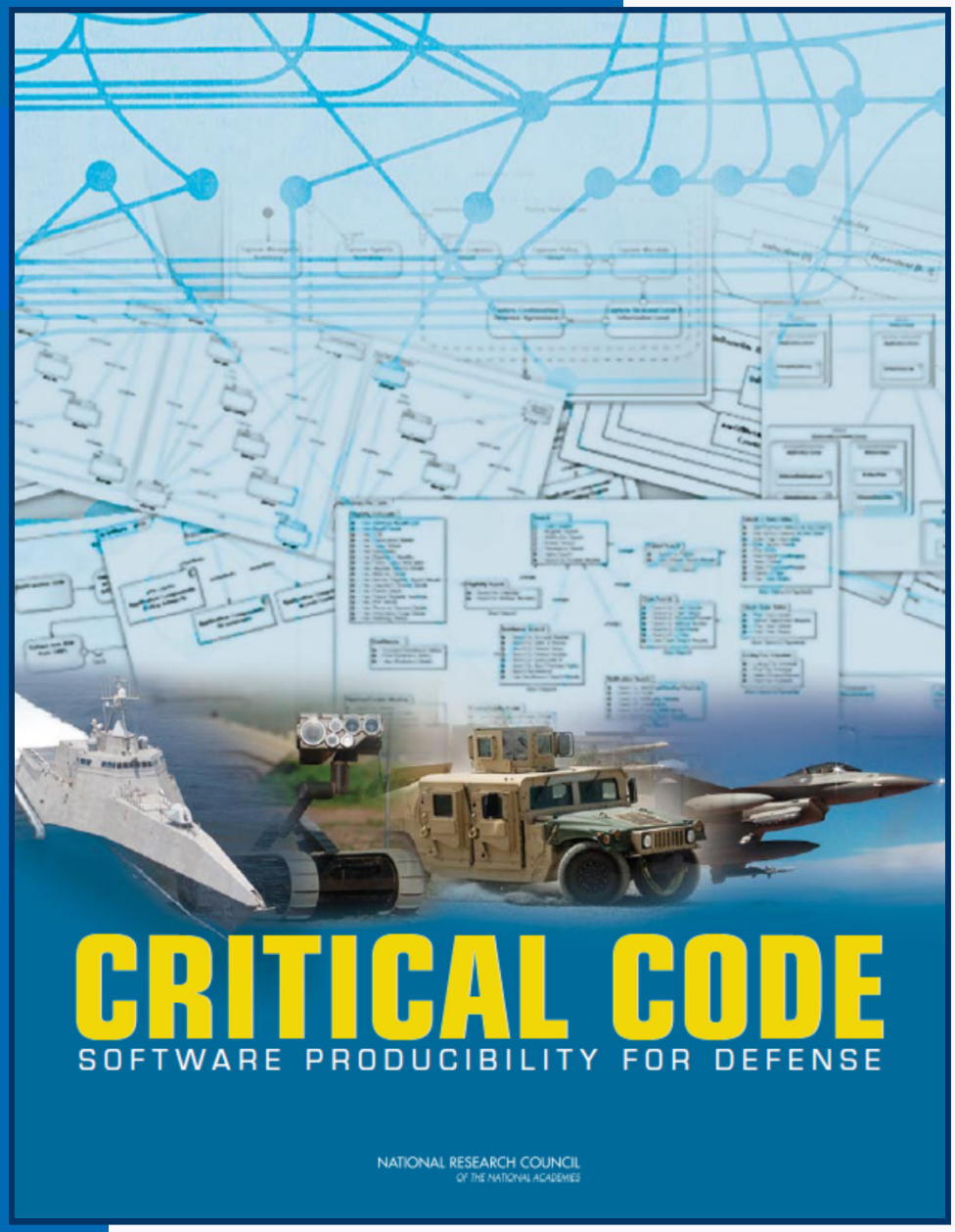


THE NATIONAL ACADEMIES
Advisers to the Nation on Science, Engineering, and Medicine

Critical Code

Software Producibility for Defense

Summary points *from the final report of the*

Committee on Advancing Software-Intensive Systems Producibility (ASISP)

William Scherlis, *Chair*

Enita Williams, Study Director
Jon Eisenberg, CSTB Director

April 2012 – SSTC, Salt Lake

National Research Council (NRC)
Computer Science and Telecommunications Board (CSTB)

The ASISP committee of the NRC

- National Research Council (NRC) ASISP Committee
  - *ASISP: Advancing Software-Intensive Systems Producibility*
  - ***Producibility:*** **the capacity to design, produce, assure, and evolve software-intensive systems in a predictable manner while effectively managing risk, cost, schedule, quality, and complexity**.
- Commissioned by the Office of the Secretary of Defense (OSD)
  - ASD(R&E) focal point, with ONR support and NSF assistance
- NRC charge to committee
  - **Assess national investment** in relevant software research
  - **Recommend improvements** to DoD software practice
  - **Examine needs** relating to DoD software research
  - **Assess research requirements** relating to software producibility

ASISP study committee

- William **Scherlis**, Carnegie Mellon University, *Chair*
- Robert **Behler**, The MITRE Corporation
- Barry W. **Boehm**, University of Southern California
- Lori **Clarke**, University of Massachusetts at Amherst
- Michael **Cusumano**, Massachusetts Institute of Technology
- Mary Ann **Davidson**, Oracle Corporation
- Larry **Druffel**, Software Engineering Institute
- Russell **Frew**, Lockheed Martin
- James **Larus**, Microsoft Corporation
- Greg **Morrisett**, Harvard University
- Walker **Royce**, IBM
- Doug C. **Schmidt**, Vanderbilt University
- John P. **Stenbit**, Independent Consultant
- Kevin J. **Sullivan**, University of Virginia

Profile

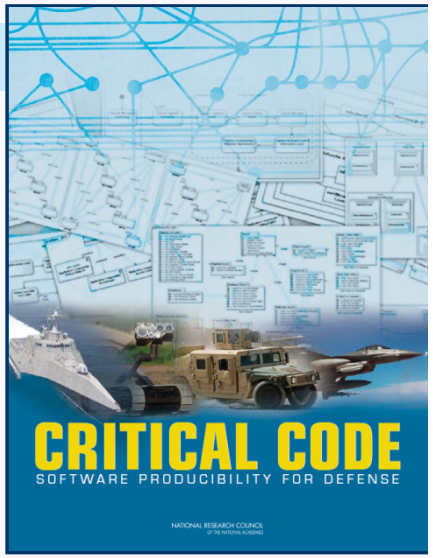
- Industry
  - Vendors
  - Primes
- Government
  - Line experience
  - Advisors
- Research
  - Academia
  - Industry

- ***CSTB Staff***
- Enita **Williams**, Study Director
- Jon **Eisenberg**, CSTB Director
- *Thanks also to*: Joan **Winston**, Lynette **Millett**, Morgan **Motto**, Eric **Whitaker**

Reviewers of the three ASISP reports

- Rick **Buskens**, Lockheed Martin ATL (*final*)
- Grady **Campbell**, Software Engineering Institute (*final*)
- William **Campbell**, BAE Systems (*final*)
- John **Gilligan**, Gilligan Group (*letter, final*)
- William **Griswold**, University of California, San Diego (*final*)
- Anita **Jones**, University of Virginia (*letter, final*)
- Annette **Krygiel**, Independent Consultant (*final*)
- Butler **Lampson**, Microsoft Corporation (*letter*)
- Steve **Lipner**, Microsoft, Inc. (*final*)
- David **Notkin**, University of Washington (*workshop, letter, final*)
- Frank **Perry**, SAIC (*final*)
- William **Press**, U Texas Austin (*final review monitor*)
- Harry D. **Raduege**, Jr., Deloitte Center for Network Innovation (*letter*)
- Alfred Z. **Spector**, Google, Inc. (*workshop, letter, final*)
- Daniel C. **Sturman**, Google, Inc. (*final*)
- John **Swainson**, CA, Inc. (*final*)
- Mark N. **Wegman**, IBM (*final*)
- John **Vu**, Boeing Corporation (*workshop*)
- Peter **Weinberger**, Google, Inc. (*workshop*)
- Jeannette **Wing**, Carnegie Mellon University (*workshop*)

Summary: The NRC *Critical Code* report



1. **Practice** – *Enhance mission capability, agility, assurance, linking*
   - **Enable incremental iterative development at arm's length**
     - Process and measurement – rethinking the practice
   - **Enable architecture leadership, interlinking, flexibility**
     - Architecture – "architecture ≈ strategy"
   - **Enable mission assurance at scale, with rich supply chains**
     - Assurance and security – evidence-based and preventive
2. **Research** – *Promote game-changers*
   - Architecture modeling and architectural analysis
   - Validation, verification, and analysis of design and code
   - Process support and economic models for assurance
   - Requirements
   - Language, modeling, code, and tools
   - Cyber-physical systems
   - Human-system interaction

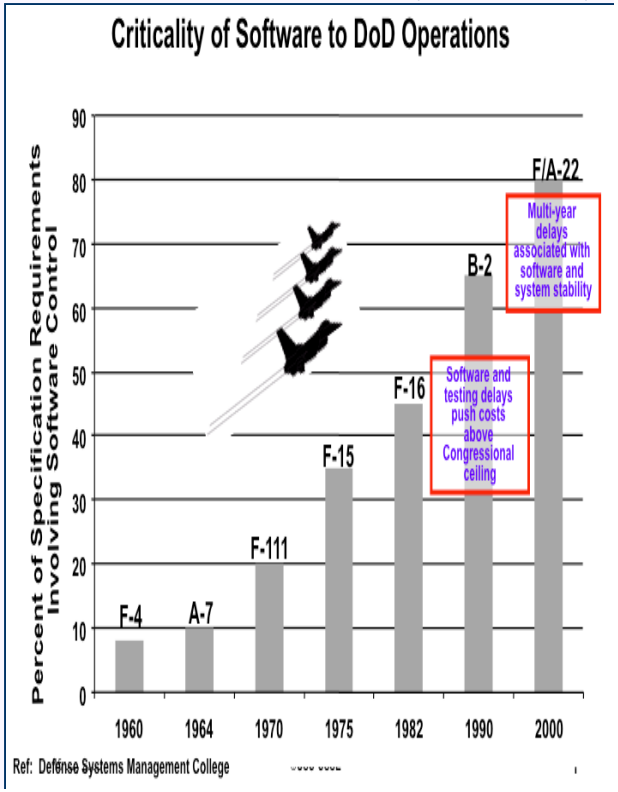
Challenge issues
- Technology leadership focal point
- Smart customer: inside expertise
- Accelerate the pipeline

3. **Leadership** – *Never relinquish the innovation lead*
   - **Recognize the unboundedness of software**
   - **Stay ahead in assurance** (cf. DSB'07)
   - **Sustain innovation and ecosystem lead**

Software has a critical role for DoD

- Mission capability embodied in software has become a unique source of strategic and military advantage
  - *Multiple DSB, NRC studies:*
    - *At the core of the ability to achieve integration and maintain mission agility is the ability of the DoD to produce and evolve **software**.*
  - *Critical code:*
    - *Software has become essential to a vast range of military system capabilities and operations.* ***Its role is deepening and broadening***
- *Roles for software:*
  - Manifesting functional **capability**
  - **Interlinking** diverse system elements
  - Enabling mission **agility**
  - Systems **development**, including M&S
- Rapid growth in extent *and* criticality

(*from DSB 2000*)



Software has broad economic significance

- Software is a principal building material for **business and infrastructure**
  - Software is now a **strategic source of competitive advantage** in sectors ranging from financial services and health care to telecom, logistics, transportation, entertainment, and utilities.
  - Software is a key building material for national infrastructure

- Disproportionate benefits from software in **economic growth**
  - ICT industries since 1995
    - ICT drives 20-15% of economic growth is US and Europe
    - ICT drives 40% of the productivity increase in Europe
  - *Most software development is outside the ICT sector*
    - ICT sector 3% of US GDP, 5% European GDP

Risks come with the benefits, 1

- Benefits and risks
  - *Benefit*: **Interlinking of systems**
    - *Risks for DoD:* Magnitude of failures, cascading failures, security challenges
  - *Benefit*: **Direct interaction by users**
    - *Risks for DoD:* More individuals can take actions with high consequence
  - *Benefit*: **Immediate enactment**
    - *Risks for DoD:* Failures and compromises can occur inside human decision loops
  - *Benefit*: **Rapid growth in capability and flexibility**
    - *Risks for DoD:* Early development and validation for architecture must be emphasized in the process
    - *Risks for DoD:* Assurance practices and tools must advance commensurably, including incremental certification and recertification

Risks come with the benefits, 2

- Benefits and risks
  - *Benefit*: **Rich supply-chain structure and geography**
    - *Risks for DoD:* Supply-chain attacks, potential for self-damaging over-reaction (provenance, ecosystems denial)
  - *Benefit*: **Modularization and rapid development**
    - *Risks for DoD:* Broad component interfaces with complex rules of engagement, assurance difficulties
  - *Benefit*: **Rich variety of accepted software ecosystems**
    - Ecosystem: conventional structure of infrastructural elements and services that are intended to be combined in a patterned way.
    - Web services stacks, iOS, Android, LAMP stack, AUTOSAR, SCADA and controls, ERP/SCM/CRM infrastructure, network
    - *Risks for DoD:* Supply chain attacks, difficulties with externalities and adoption, difficulties with enhanced compliance practices

Software is not “maturing to a plateau”

1958.

```
10      DO 11 I = 1, 10
11      A(I) = I*N(I)
```

The tide of abstraction continues to rise

NATIONAL PHYSICAL LABORATORY

TEDDINGTON, MIDDLESEX, ENGLAND

PAPER 2-3

AUTOMATIC PROGRAMMING
PROPERTIES AND PERFORMANCE OF
FORTRAN SYSTEMS I AND II

by

J. W. BACKUS

To be presented at a Symposium on
The Mechanization of Thought Processes,
which will be held at the National Physical Laboratory, Teddington, Middlesex, from 24th-27th November 1958. The papers and the discussions are to be published by H.M.S.O. in the Proceedings of the Symposium. This paper should not be reproduced without the permission of the author and of the Secretary, National Physical Laboratory.

Software is not “maturing to a plateau”

- The myth of the plateau
  - We are not at a plateau or near a plateau in overall software capability or technology for software producibility
- Software has intrinsic **unboundedness**
  - Few physical limits on scale and complexity
  - Only human intellectual limits and mathematical limits on algorithms
  - New software-manifest capabilities continue to emerge
    - Order-of-magnitude leaps – constant disruption
  - Enabled by a steady pace of technological breakthroughs
    - Models, languages, tools, and practices
  - Leveraged through software ecosystems and infrastructure
- The software story is just beginning….

Software is not “maturing to a plateau”

- The myth of the plateau
  - We are not at a plateau or near a plateau in overall software capability or technology for software producibility

- DoD has a critical role in assuring its software future
  - ***DoD cannot rely on industry alone*** *to address the long-term software challenges particular to defense.*
    - ***DoD’s needs will not be sufficiently met*** *through a combination of demand-pull from the military and technology-push from the defense or commercial IT sectors.*
  - ***DoD must take immediate action*** *to reinvigorate its investment in software producibility research*
    - *Undertake* ***research*** *programs in seven identified areas*
    - *ASD(R&E) should regularly undertake an identification of areas of* ***technological need*** *related to software producibility where the DoD has “leading demand” and where accelerated progress is needed*

Further consequences of unboundedness

- Software engineering and other engineering
  - **A relatively *much larger portion* of overall software engineering effort is creating *innovative* functionalities, as compared with other engineering disciplines**
    - Hence: Higher levels of *engineering risk*
    - Hence: Greater need to decouple engineering risk from program risk
- Sustaining capability
  - **Mere presence as a software user** requires keeping pace with rapid ongoing innovation and improvement to practices
- The imperative for software leadership
  - *There is strategic value to DoD in the US sustaining its leadership in software producibility -- compared with other industries that have moved offshore*

Deliberation and challenges – Software Myths

- Long-standing **incorrect folklore** regarding defense software producibility (*digested from the report*)

1. DoD software producibility challenges are predominantly challenges of **management** and process but **not of technology**.

2a. DoD and contractors can **rely on industry to innovate** at a rate fast enough to solve the DoD's hard technical problems and to stay ahead of its adversaries.

2b. Regardless, there is **sufficient software research already underway** through NSF and other sponsors.

3. Software technology is **approaching a plateau**, which diminishes the benefit from investing in technology innovation.

Deliberation and challenges – Software Myths

- Long-standing **incorrect folklore** regarding defense software producibility (*digested from the report*)

4. We will **never create perfectly reliable and secure** software, so we should focus primarily on provenance—trusted sources—rather than attempting to achieve assurance directly.

5. **Earned value management** approaches based on code accumulation are a sufficient basis for managing software development programs, including incremental iterative development.

SDP+HCSS relative to total NITRD

PITAC 1999:

Finding: **The Nation is underinvesting in fundamental software research.**

Recommendation: **Make fundamental software research one of the Nation's highest R&D priorities.**

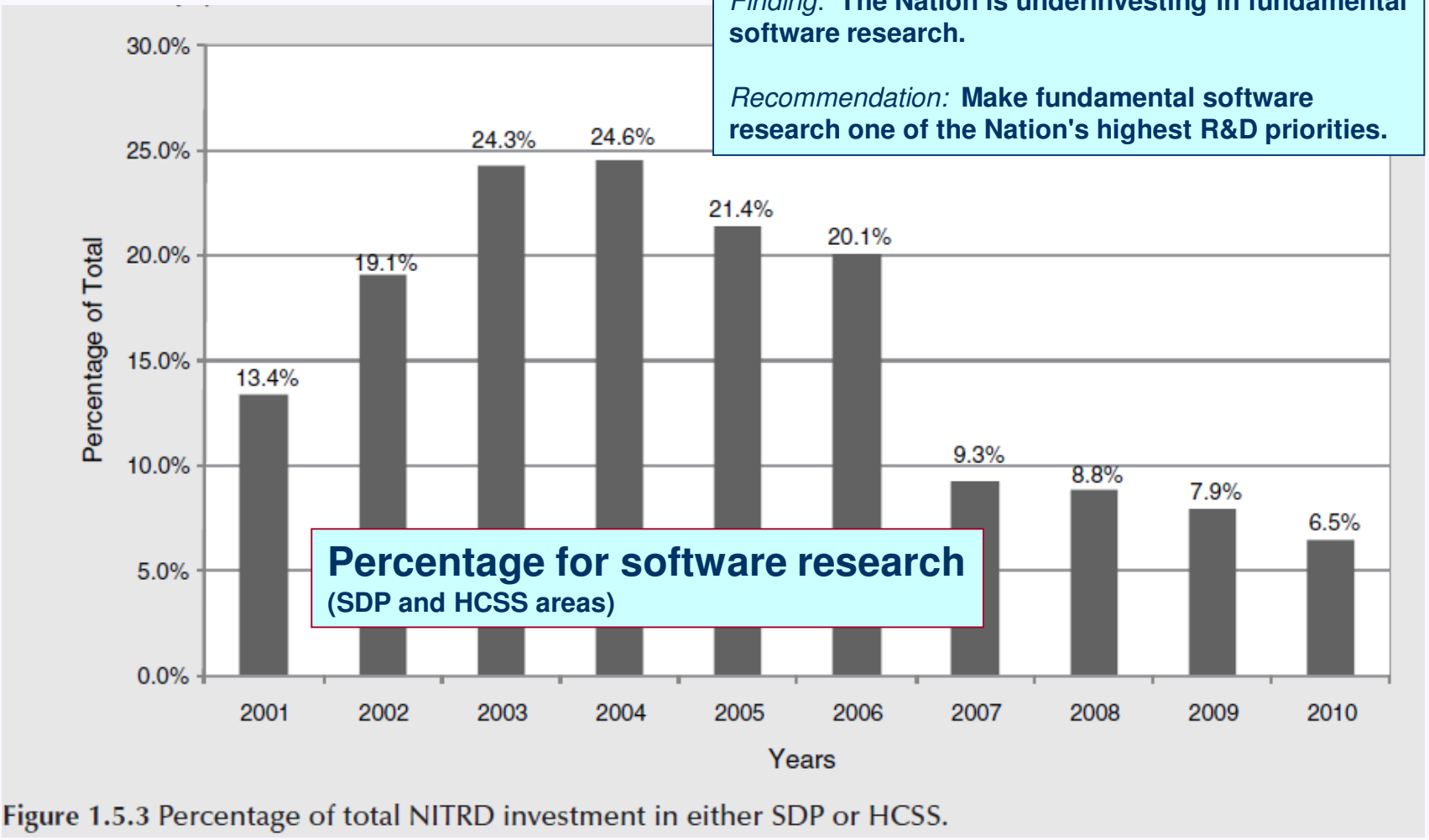


Figure 1.5.3 Percentage of total NITRD investment in either SDP or HCSS.

Outline

- Task and prior reports
- Committee, process, background
- Economic argument
- **Areas of practice**
  - **Process and measurement**
  - Architecture
  - Software expertise
  - Assurance and security
- Topics of research
- Next steps

Chapter 2 of the report

Incremental and iterative software dev't practices

- ***Innovative engineering does not necessarily entail increased programmatic risk***
  - The decoupling is achieved through **incremental identification and mitigation of engineering uncertainties**.
    - Incremental Iterative Development (IID)
  - For defense software, the **uncertainties** relate to (1) unprecedented concepts, (2) scale and interlinking, (3) software in systems engineering, and (4) arm's-length contractor relationships.
- Conclusions regarding enablers of IID
  - Improved **measurement** capability for the attributes that matter
    - Supported by **accumulation of high-quality data** regarding project management experience and technology choices.
  - Extensions to **earned value management models**
    - Including evidence of feasibility and time-certain development.
  - Supplementing the prescription of DoDI 5000.02
    - Better support for ongoing management of engineering risks

Outline

- Task and prior reports
- Committee, process, background
- Economic argument
- **Areas of practice**
  - Process and measurement
  - **Architecture**
  - Software expertise
  - Assurance and security
- Topics of research
- Next steps

Chapter 3 of the report

Assert architecture leadership

- **Architecture** has a pivotal role in complex innovative systems:
  - Embodiment of planning for **flexibility**—defining and encapsulating areas where innovation/change are anticipated—and hence agility.
  - Determiner of **quality attributes**—security, performance, resilience, etc.
  - Enabler for **product lines** (time and space) and **interlinking** of systems and **SoS** / **COE** / **net-centric** / **ULS** structures generally.
- *DoD needs to play an active role in **software architecture**.*
  - Software architecture
    - Definition: *The structure or structures of the system, which comprise software components, the externally visible properties of those components, and the relationships among them*.
    - Good architecture entails a minimum of engineering commitment that yields maximum value.
    - Architecture includes the earliest and often most important design decisions – those that are most difficult to change later.

Assert architecture leadership

- Architecture and requirements for innovative systems
  - **Consideration of architecture and quality attributes may best precede commitment to specific functionality.**
  - Architecture is profoundly influenced by precedent
    - Small changes can open and close opportunities to exploit rich **ecosystems**, greatly influencing cost, risk, and supply chain structure
  - Architecture design is an engineering activity that is **separate from ecosystems certification** and other standards-related policy setting.
- Conclusions regarding architecture
  - **An early focus on architecture is essential for systems with innovative functional or quality requirements.**
  - **Architecture practice, as seen in industry, is sufficiently mature for DoD to adopt**
  - **Follow architecture-driven acquisition strategies**
    - **Support early and continuous validation of architectural decisions**

Outline

- Task and prior reports
- Committee, process, background
- Economic argument
- **Areas of practice**
  - Process and measurement
  - Architecture
  - **Software expertise**
  - Assurance and security
- Topics of research
- Next steps

Chapter 2 of the report

There is insufficient DoD-aligned software expertise

- Conclusions regarding DoD-aligned expertise
  - The DoD has a **growing need** for software expertise
    - It is not able to meet this need through intrinsic DoD resources.
    - Nor is it able to fully outsource this requirement to DoD primes.
  - The **DoD needs to be a smart software customer**
    - Particularly for large-scale innovative software-intensive projects.

- *No recommendation offered*

Outline

- Task and prior reports
- Committee, process, background
- Economic argument
- **Areas of practice**
  - Process and measurement
  - Architecture
  - Software expertise
  - **Assurance and security**
- Topics of research
- Next steps

Chapter 4 of the report

Adopt a strategic approach to software assurance

- Finding from DSB2007, reiterated in Critical Code
  - It is an essential requirement that the United States maintain advanced capability for "test and evaluation" of IT products. Reputation-based or trust-based credentialing of software ("provenance") needs to be augmented by direct, artifact-focused means to support acceptance evaluation.
- Assurance challenges
  - Inadequate + costly **legacy** approaches (inspection, sampled tests)
  - Newly rich and globally diverse **supply chains**, with arms-length relationships
  - Assurance requirements can dramatically limit systems capability
  - **High consequences** due to roles in war-fighting and protection of lives and assets
  - Failure to maintain a lead in assurance confers advantage to adversaries
- Assurance opportunities
  - Technical advances and potential for preventive/evaluative practices
    - *Evidence production*
    - *Architecture analysis*
    - *Isolation / encapsulation*
    - *Configuration management*
- Conclusion
  - DoD must directly foster advanced software practice and tools for highly assured high capability systems -- nobody is doing this for DoD

Fears from 2003

COMPUTERWORLD
Careers

In Depth | Reviews | White Papers | Newsletters | IT Jobs

Google™ Custom Searc

Hidden malware in offshore products raises concerns

By Mark Willoughby
September 15, 2003 12:00 PM ET

Recommended (4) Digg Twitter Share/Email

Computerworld - *"You've go to be a little paranoid to survive in this business." -- Andrew S. Grove, chairman and founder, Intel Corp., circa 1980*

The extreme difficulty in discovering back doors hidden deep within a complex application, buried among numerous modules developed offshore in a global software marketplace, is forcing those assigned to protect sensitive national security information to take defensive actions.

The threat of hidden Trojan horses and back doors surfaced this summer when the governments of the U.S. and China announced plans to strengthen national security policies covering information processed by applications written in the global software marketplace. The private sector joined the fray with the August announcement of the File Signature Database, which will use hash values to protect software integrity from malicious additions (see story).

The National Security Agency's information assurance director, Daniel Wolf, in testimony before the House Select Committee on Homeland Security's cybersecurity subcommittee in July, called for a federal lab

White Papers & Webcasts

LIVE WEBCAST: Maximize ROI for Web Applic
LIVE Feb 23, 2010 02:00 PM ET

LIVE WEBCAST: BIRT for the Enterprise: Solv
Visualization Needs with Open Source BI
LIVE Feb 25, 2010 02:00 PM ET

26

“Foreign influence” on software – DSB 2007

- *Provenance is a poor surrogate for direct evaluation*
- *We need to be better at understanding our own code*

RECOMMENDATION: DoD should provide incentives to industry to produce higher quality code.

There will never be a single set of best practices, testing tools, methodologies, or development process that works for all vendors -- nor is that desirable, since

Promote the Use of Automated Tools in Product Development

Tools for the detection of vulnerabilities continue to improve and proliferate. Different vendors will choose to use different tools; no one tool is right for every company. The DoD may well have an interest in determining how good the tools are that vendors use, but DoD should not be in the position of dictating particular

Report of the
Defense Science Board Task Force
on
Mission Impact of Foreign Influence on DoD Software

September 2007

Office of the Under Secretary of Defense
For Acquisition, Technology, and Logistics
Washington, D.C. 20301-3140

27

Challenges of software assurance

- **Current technical approaches to software assurance are inadequate.**
  - ***Assurance***
    - *Definition: Human judgment regarding functional and quality attributes (reliability, safety, security, etc.)*
  - Current technical approaches need to be augmented
    - **Costs** range from 30-50% for typical major projects
    - **Recertification** is very costly
    - Conventional testing and inspection techniques are **inadequate** for modern software-intensive systems development
  - Cannot be achieved entirely through *post hoc* acceptance evaluation
    - Quality and security are **designed and built in**, not "tested in"
    - **Not analogous** to reliability models for physical systems

There is progress in assurance practices and tools

- Conclusions for assurance
  - Assurance is facilitated by recent advances in **diverse aspects** of software engineering practice and technology.
    - Modeling, analysis, tools and environments, traceability, programming languages, and process support.
  - **Early** engineering choices strongly influence feasibility of achieving high assurance.
    - Successful approaches involve a diverse set of evaluative and preventive techniques
  - New models support the **simultaneous** creation of assurance-related **evidence** with ongoing development effort
    - Create and sustain ***chains of evidence*** that link software-related artifacts including design, functional and quality-attribute requirements, code, test cases, inspection reports, analysis, simulation, models, etc.
  - Institute effective **incentives** for preventive software assurance practices and production of evidence across the lifecycle.
    - Do this throughout the supply chain

Outline

- Task and prior reports
- Committee, process, background
- Economic argument
- Areas of practice
- **Topics of research**
  - **Seven technology areas**
  - **Four considerations**
  - **Reinvigoration plan**
- Next steps

Chapter 5 of the report

Reinvigorate DoD software engineering research

- Focus research effort in seven technology areas that directly enable producibility improvements
  1. **Architecture modeling and architectural analysis**
     Goals:
     (1) Early validation for architecture decisions
     (2) Architecture-aware systems management
     - Including: Rich supply chains, ecosystems, and infrastructure

     (3) Component-based development
     - Including: Architectural designs for particular domains.

  2. **Validation, verification, and analysis of design and code**
     Goals:
     (1) Effective evaluation for critical quality attributes
     (2) Components in large heterogeneous systems
     (3) Preventive methods to achieve assurance
     - Including: Process improvement, architectural building blocks, programming languages, coding practice, etc.

  3. **Process support and economic models for assurance**
     Goals:
     (1) Enhanced process support for assured software development
     (2) Models for evidence production in software supply chains
     (3) Application of economic principles to process decision-making

continued…

Reinvigorate DoD software engineering research

- Focus research effort in seven technology areas that directly enable producibility improvements

4. **Requirements**

   Goals:

   (1) Expressive models, supporting tools for functional and quality attributes

   (2) Improved support for traceability and early validation

5. **Language, modeling, coding, and tools**

   Goals:

   (1) Expressive programming languages for emerging challenges

   (2) Exploit modern concurrency: shared-memory and scalable distributed

   (3) Developer productivity for new development and evolution

6. **Cyber-physical systems**

   Goals:

   (1) New conventional architectures for control systems

   (2) Improved architectures for embedded applications

7. **Human-system interaction**

   Goal:

   (1) Engineering practices for systems in which humans play critical roles

   (*This area is elaborated in another NRC report*)

Considerations in identifying research topic areas

(1) **Significant potential value** for DoD software producibility
- Process and measurement, architecture, and assurance (chapters 2, 3, 4)

(2) **Feasible progress** in a well-managed research program
- Well-managed with respect to “Heilmeier Questions”
- There is past success in software research, now well documented

(3) **Not addressed sufficiently** by other federal agencies
- Primarily other NITRD-coordinated agencies

(4) **Might not otherwise develop** at a sufficient pace
- In industry or through research sponsored elsewhere

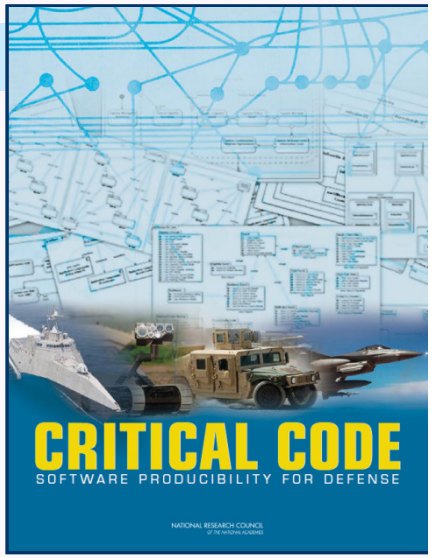
Software engineering research revisited

- Challenges and success influences
  - Software engineering is **maturing as a research discipline**.
    - Improved research methods and lower risk in technology transition
    - Data-rich environment
    - Facilitating more satisfactory responses to the Heilmeier Questions
  - **Diffusion pathways** are complex, with variability of timescale.
    - Some results can readily transfer to DoD practice
    - Others, often most significant, take longer and are more indirect
  - Novelty is often more about **timeliness**.
    - Readiness (infrastructure, exponentials) rather than technical novelty
    - What are ideas whose time has come? (E.g., thin/rich client, utility/cloud)
  - We can accept non-quantitative means to **assess progress**.
    - Often focus of research is on developing such measures
    - *Example*: how to quantify the benefits of strong typing?

Outline

- Task and prior reports
- Committee, process, background
- Economic argument
- Areas of practice
- Topics of research
  - Seven technology areas
  - Four considerations
  - Reinvigoration plan
- **Next steps**

Summary: The NRC *Critical Code* report



1. **Practice** – *Enhance mission capability, agility, assurance, linking*
   - **Enable incremental iterative development at arm's length**
     - Process and measurement – rethinking the practice
   - **Enable architecture leadership, interlinking, flexibility**
     - Architecture – "architecture ≈ strategy"
   - **Enable mission assurance at scale, with rich supply chains**
     - Assurance and security – evidence-based and preventive
2. **Research** – *Promote game-changers*
   - Architecture modeling and architectural analysis
   - Validation, verification, and analysis of design and code
   - Process support and economic models for assurance
   - Requirements
   - Language, modeling, code, and tools
   - Cyber-physical systems
   - Human-system interaction

Challenge issues
- Technology leadership focal point
- Smart customer: inside expertise
- Accelerate the pipeline

3. **Leadership** – *Never relinquish the innovation lead*
   - **Recognize the unboundedness of software**
   - **Stay ahead in assurance** (cf. DSB'07)
   - **Sustain innovation and ecosystem lead**